APSIPA Transactions on Signal and Information Processing, 2024, 13, e404

Original Paper

End-to-End Singing Transcription Based on CTC and HSMM Decoding with a Refined Score Representation

Tengyu Deng[1], Eita Nakamura[2*], Ryo Nishikimi[3] and Kazuyoshi Yoshii[1]

[1] *Graduate School of Informatics, Kyoto University, Japan*
[2] *Graduate School of Information Science and Electrical Engineering, Kyushu University, Japan*
[3] *NTT Communication Science Laboratories, Japan*

ABSTRACT

This paper describes an end-to-end automatic singing transcription (AST) method that translates a music audio signal containing a vocal part into a symbolic musical score of sung notes. A common approach to sequence-to-sequence learning for this problem is to use the connectionist temporal classification (CTC), where a target score is represented as a sequence of notes with discrete pitches and note values. However, if the note value of some note is incorrectly estimated, the score times of the following notes are estimated incorrectly and the metrical structure of the estimated score collapses. To solve this problem, we propose a refined score representation using metrical positions of note onsets. To decode a musical score from the output of a deep neural network (DNN), we use a hidden semi-Markov model (HSMM) that incorporates prior knowledge about musical scores and temporal fluctuation in human performance. We show that the proposed method achieves the state-of-the-art performance and confirm the efficacy of the refined score representation and the decoding method.

*Corresponding author: nakamura@inf.kyushu-u.ac.jp

Received 08 March 2024; revised 26 June 2024; accepted 22 August 2024
ISSN 2048-7703; DOI 10.1561/116.20240016

1 Introduction

Automatic singing transcription (AST) is one of the most fundamental tasks in the field of music information retrieval (MIR). Its ultimate goal is to estimate a human-readable musical score of the vocal part from a music audio signal. This technique is useful in many downstream tasks such as music search, query-by-humming [12, 24], interpretable emotion recognition [2], and score-guided singing voice separation [4, 5]. There are two major types of representations in the output of AST: a piano-roll (MIDI) representation and a musical score representation. In the piano-roll representation, the onset and offset times of sung notes are measured in frames or seconds. In a symbolic musical score, they are represented in *score times* measured in quantized time units such as fractions of beats.

Most studies on AST have focused on *audio-to-MIDI* transcription [10, 11, 13, 30, 32], where each sung note is represented by a semitone-level pitch, onset time, and offset time in seconds. Generally, a contour of fundamental frequencies (F0s) in Hz is estimated with a deep neural network (DNN) and a note tracking method is then used to segment the F0 contour to a series of notes [10, 32]. One can apply rhythm transcription methods for converting a piano roll to a musical score by quantizing the onset and offset times [18, 27]. However, such a cascading approach can suffer from the accumulation of errors [3].

A few studies have dealt with *audio-to-score* singing transcription. Nishikimi *et al.* [21, 22] attempted to obtain a musical score by quantizing certain framewise features by using the ground-truth or pre-estimated beat times. These methods require time-aligned ground-truth data such as beat times in the training process. However, creating the time-aligned annotations requires much effort, and currently available datasets with such annotations are limited both in size and variation. To solve this problem, it is desirable to develop an end-to-end AST method that can be trained directly with non-aligned pairs of audio signals and musical scores.

The connectionist temporal classification (CTC) [7] is a common technique to deal with non-aligned pairs of sequential data. It was first introduced in the context of end-to-end audio-to-text automatic speech recognition (ASR). It works as a loss function between two sequences with different lengths (e.g., a sequence of frame-level acoustic features and a sequence of words or phonemes) by internally estimating the monotonic alignment between them. This enables the effective use of non-aligned audio-text pairs for training a DNN.

Recently, several studies have applied CTC for audio-to-score automatic music transcription (AMT) [1, 26, 25]. These studies typically used a score representation where note values (NVs), i.e., score-written lengths of notes, represent the temporal information. We call this type of score representation an NV-based representation. However, the NV-based representation has some

essential problems. First, it is questionable whether the CTC-based learning is effective for recognizing the temporal information represented by NVs. This is because the original CTC was designed for estimating instantaneous symbols (e.g., phonemes) associated with local acoustic features, whereas it is necessary to aggregate acoustic features from a wide time region to estimate NVs representing duration information. Second, the quality of the estimated score is sensitive to minor errors of NVs [19]. If the note value of some note is incorrectly estimated, then this affects the score times of all the following notes and harms the metrical structure (such as the positions of barlines) of the score, which is musically important.

To solve these problems, we propose a metrical-position-based (MP-based) score representation for CTC-based AST. The MP of a note represents the position of its onset time relative to a barline. The MP-based representation is suited for capturing metrical structure and retains the information in the NV-based representation. Our method aims to convert a sequence of acoustic features into a sequence of sung notes, each represented by a semitone-level pitch and an MP. Specifically, we train a convolutional recurrent neural network (CRNN) predicting the frame-level probabilities of these symbols and their boundaries. To improve the performance in the decoding process, we also propose a hidden semi-Markov model (HSMM) that incorporates prior knowledge about musical scores and temporal fluctuation in human performance.

The rest of the paper is organized as follows. We formalize musical score representations in Section 2 and review related studies in Section 3. The proposed method is presented in Section 4 and the evaluation results are reported in Section 5. We conclude our study in Section 6.

2 Musical Score Representations

This section explains three different representations of musical scores relevant in this study. Musical pieces are assumed to have the time signature of 4/4. The finest time units on the musical score are called *tatums*, which are in general dependent on individual pieces. For the sake of mathematical formulation, we assume a unified tatum unit, which is treated as a hyperparameter of the method. If the tatums are assumed to be Q times finer than the beats, then tatums will correspond to the $4Q$-th notes. Each measure thus contains $B = 4Q$ tatums, and the MP of each tatum within the measure is represented as an index between 0 and $B - 1$. In this paper we mainly consider the case $Q = 4$ and $B = 16$. In this condition, triplet notes or notes shorter than 16th notes cannot be represented, and we consider that the NVs of such notes are rounded to the closest integer multiples of tatum intervals. This problem can be partially addressed by using a finer-grained tatum unit, as discussed in Section 5.2.3.

We represent the musical score of a vocal melody as a sequence of notes $\mathbf{Y} = (y_n)_{n=1}^N = (\tau_n, l_n, p_n)_{n=1}^N$, where N is the number of notes. An integer τ_n represents the onset score time of the n-th note in units of tatums (we usually assume that $\tau_1 = 0$). The NV l_n is represented as a multiple of tatums. An integer p_n represents the pitch in units of semitones and we use the conventional MIDI note number extended to include the rest: $p_n \in \{0, 1, \ldots, 128\}$, $p_n = 128$ indicates a rest, and if $p_n < 128$ it represents a pitch (e.g., C4 = 60).

As data representation used for training DNNs, we can use some variants of score representation $\mathbf{Y}'$. As explained in the following, the definition of each symbol and the sequential length vary according to the representation.

2.1 Tatum-Level Representation

In our previous work, we used the tatum-level representation [3]. In detail, we consider a sequence of tatum-level score fragments $\mathbf{Y}^{\text{tatum}} = (y_m^{\text{tatum}})_{m=1}^M = (b_m^{\text{tatum}}, p_m^{\text{tatum}}, o_m^{\text{tatum}})_{m=1}^M$, where M is the number of tatums, $b_m^{\text{tatum}} = m \bmod B \in \{0, \ldots, B-1\}$ is the MP of the tatum, $p_m^{\text{tatum}} \in \{0, \ldots, 128\}$ is the pitch, and $o_m^{\text{tatum}} \in \{0, 1\}$ is the onset flag indicating the absence (0) or presence (1) of a note/rest onset. Figure 1(b) shows an example of the tatum-level representation. $\mathbf{Y}^{\text{tatum}}$ can be obtained from $\mathbf{Y}$ by

$$p_m^{\text{tatum}} = p_n, \quad \tau_n \le m < \tau_{n+1}, \tag{1}$$

$$o_m^{\text{tatum}} = \begin{cases} 1, & \exists n, \text{ s.t. } m = \tau_n, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

An issue of this representation is that the inverse process to reconstruct $\mathbf{Y}$ from $\mathbf{Y}^{\text{tatum}}$ requires the following condition:

$$b_{m+1}^{\text{tatum}} - b_m^{\text{tatum}} \equiv 1 \bmod B, \qquad m = 1, \ldots, M. \tag{3}$$

Since the output sequence directly obtained by a DNN may not meet this condition in the inference step, an HSMM was used to impose the condition [3].

2.2 Note-Level Representations

As note-level representations, we propose the NV-based and MP-based representations, both of which can be considered as a reduced version of the original score representation $\mathbf{Y}$.

2.2.1 NV-Based Representation

We define the NV-based representation as $\mathbf{Y}^{\text{NV}} = (y_n^{\text{NV}})_{n=1}^N = (l_n^{\text{NV}}, p_n^{\text{NV}})_{n=1}^N$, where N is the number of notes, l_n^{NV} is the NV of the n-th note measured in

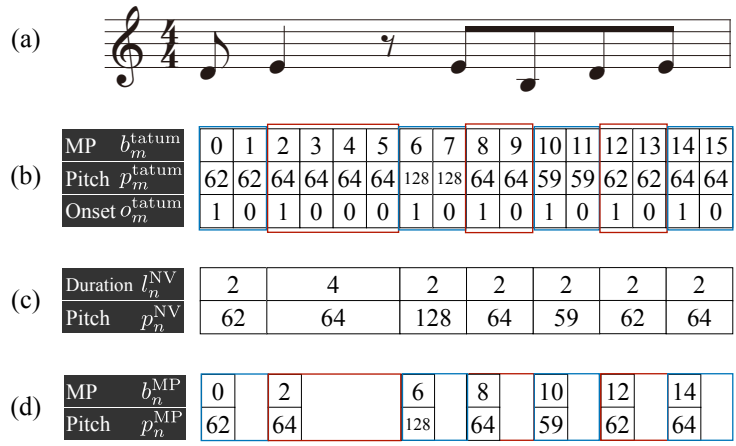

Figure 1: Musical score representations. (a) Musical score. (b) Tatum-level representation [3]. (c) NV-based note-level representation (conventional). (d) MP-based note-level representation (proposed).

tatums, and $p_n^{\text{NV}} \in \{0, \ldots, 128\}$ is its pitch. Figure 1(c) shows an example of the NV-based representation.

The NV-based representation can be simply obtained from score $\mathbf{Y}$:

$$l_n^{\text{NV}} = l_n, \quad n = 1, \ldots, N, \tag{4}$$

$$p_n^{\text{NV}} = p_n, \quad n = 1, \ldots, N. \tag{5}$$

If we assume $\tau_n = 0$, then the process to obtain $\mathbf{Y}$ from $\mathbf{Y}^{\text{NV}}$ is also straightforward:

$$\tau_n = \begin{cases} 0, & n = 1, \\ \sum_{n'=1}^{n-1} l_{n'}^{\text{NV}}, & n = 2, \ldots, N, \end{cases} \tag{6}$$

$$l_n = l_n^{\text{NV}}, \quad n = 1, \ldots, N, \tag{7}$$

$$p_n = p_n^{\text{NV}}, \quad n = 1, \ldots, N. \tag{8}$$

In reality, musical notes often continue over barlines via ties and their NVs can be arbitrarily large. To define a finite vocabulary used by a DNN, we consider an upper bound B of NVs so that $l_n^{\text{NV}} \in \{1, \ldots, B\}$. Notes longer than B in the data are formally split into non-tied notes for training the DNNs. This makes the NV-based representation contain less information than the original musical score. Since such long notes are relatively rare in reality, the NV-based representation is almost equivalent to the complete score representation $\mathbf{Y}$.

2.2.2 MP-Based Representation

To capture the metrical structure, we define the MP-based representation as $\mathbf{Y}^{\mathrm{MP}} = (y_n^{\mathrm{MP}})_{n=1}^N = (b_n^{\mathrm{MP}}, p_n^{\mathrm{MP}})_{n=1}^N$, where N is the number of notes, $b_n^{\mathrm{MP}} \in \{0, \ldots, B-1\}$ is the onset MP of the n-th note, and $p_n^{\mathrm{MP}} \in \{0, \ldots, 128\}$ is the pitch. Figure 1(d) shows an example of the MP-based representation. One can obtain the MP-based representation $\mathbf{Y}^{\mathrm{MP}}$ from $\mathbf{Y}$ by

$$b_n^{\mathrm{MP}} = \tau_n \bmod B, \quad n = 1, \ldots, N, \tag{9}$$

$$p_n^{\mathrm{MP}} = p_n, \quad n = 1, \ldots, N. \tag{10}$$

The reconstruction process is slightly more complicated than that for the NV-based representation. Similar as in Section 2.2.1, we first assume that $l_n \leq B$, and we define Δ as the NV indicated by two consecutive MPs:

$$\Delta(b, b') = \begin{cases} B, & b = b', \\ (b' - b) \bmod B, & b \neq b'. \end{cases} \tag{11}$$

Then, we can reconstruct the musical score $\mathbf{Y}$ from $\mathbf{Y}^{\mathrm{MP}}$ by

$$l_n = \Delta(b_n^{\mathrm{MP}}, b_{n+1}^{\mathrm{MP}}), \quad n = 1, \ldots, N-1 \tag{12}$$

$$p_n = p_n^{\mathrm{MP}}, \quad n = 1, \ldots, N, \tag{13}$$

$$\tau_n = \begin{cases} 0, & n = 1, \\ \sum_{n'=1}^{n-1} l_{n'}, & n = 2, \ldots, N. \end{cases} \tag{14}$$

Note that the NV of the last note l_N cannot be determined, and we need some extra information (such as the offset MP of the last note) to determine it. Additionally, this representation fails to deal with notes longer than B as in the NV-based representation.

As explained in the Introduction, the NV- and MP-based representations have an important difference regarding the robustness of score times against a slight change in the note-level symbols, which can often be contained in a DNN's estimation. In the NV-based representation, (6) explicitly indicates that a slight change in the NV of a note affects the score times and MPs of all the following notes. In the MP-based representation, however, a slight change in the MP of the n-th note only affects the NVs of the n-th and $(n+1)$-th notes by (11), but the score times of the other notes are unaffected. Since the metrical structure is an important factor that influences the perceptual quality of a musical score, the MP-based representation is expected to be more suitable for AST.

3 Related Work

We here review AST and AMT methods based on the cascading and end-to-end approaches.

3.1 Cascading Approach

Some studies proposed to combine frame-level F0 estimation and note tracking for audio-to-MIDI transcription [14, 15]. DNNs have recently been used to improve the former task [10, 13, 32]. Nishikimi *et al.* [21] proposed an HSMM to quantize an F0 contour into a sequence of musical notes. Hsu and Su [11] attempted to directly transcribe an audio signal into note events to overcome the error accumulation problem of the cascading approach. To improve the performance of audio-to-MIDI transcription, the CTC loss was used in addition to the standard framewise cross-entropy loss [30].

In the context of AST that aims to estimate musical scores, rhythm transcription methods have been explored to quantize a piano-roll representation into a musical score [18, 19, 27]. Nishikimi *et al.* [22], for example, proposed a hybrid DNN-HSMM model to circumvent the dependency on the accuracy of F0 estimation, in the same way as the DNN-HMM approach to ASR. In general, the MPs or NVs of musical notes are used as rhythmic information. Studies using HMMs have shown that the MP-based representation generally performs better than the NV-based representation because metrical structure cannot be represented in the latter [19].

3.2 End-to-End Approach

Several studies have recently investigated end-to-end AMT to address the error accumulation problem of the cascading approach. Inspired by the success of end-to-end ASR, the attention mechanism and/or the CTC have been used for learning the audio-to-score mapping using non-aligned data. Some AMT methods [1, 26] used the CTC with the NV-based note-level representation. However, these methods have been shown to work only on music signals synthesized from MIDI data at the proof-of-concept level. In another attempt, an encoder-decoder model with an attention mechanism was used with the tatum-level score representation [23], or the NV-based note-level representation [20]. In the end-to-end training, the internal attention matrix is regularized to have the monotonic and regular frame-to-tatum alignment. Such regularization, however, often prevents the iterative optimization method from finding the correct alignment because the attention matrix is not necessarily monotonic in early epochs of non-regularized training.

4 Proposed Method

This section describes the proposed audio-to-score AST method that uses the CTC with the NV- or MP-based score representation.

4.1 Problem Specification

The input is a frame-level sequence of acoustic features $\mathbf{X} = (\mathbf{x}_t)_{t=1}^{T}$, where T is the number of frames. Each $\mathbf{x}_t \in \mathbb{R}^{2\times D}$ is obtained at time t by stacking the mel-spectrum of a music signal and that of a singing voice separated with a voice separation method (specifically we use Open-Unmix [28]), where D is the number of mel-frequency bins. We choose the mel-spectrum as the input feature because it can well represent the features of human voice and thus is widely used in the field of ASR. The output is a score representation $\mathbf{Y} = (y_n)_{n=1}^{N}$ as explained in Section 2.

In the training phase, we aim to train a DNN using paired data of the acoustic features $\mathbf{X}$ and the ground-truth representation $\mathbf{Y}$. In the test phase, given only $\mathbf{X}$ as input, we aim to estimate $\mathbf{Y} = (y_n)_{n=1}^{N}$ and reconstruct a musical score of sung notes. The input length must be longer than the output length ($T > N$), which always holds in AST.

4.2 End-to-End Training

We train a DNN that estimates the posterior probabilities of symbols at the frame level using the CTC loss. More specifically, we use the multi-label CTC (MCTC) loss [31] for joint estimation of multiple attributes. In the following subsections, we formulate two DNNs using the NV- and MP-based score representation, respectively.

4.2.1 NV-Based Representation

We explain the CTC-based training using the acoustic features $\mathbf{X}$ and the ground-truth NV-based representation $\mathbf{Y} = (\mathbf{L}, \mathbf{P})$. Let B be the maximum value of NV and $I = 129$ be the size of the pitch vocabulary. To represent a frame-level sequence of symbols, the extra blank symbol "$*$" working as a wild card is introduced. A blank symbol represents a continuation of the previous symbol in the frame-level sequence. Let $\bar{\mathbf{Y}} = (\bar{y}_t)_{t=1}^{T}$ be a *frame-level* sequence of symbols such that $\mathbf{Y} = \mathcal{M}(\bar{\mathbf{Y}})$, where $\bar{y}_t \in \{(l,p)\}_{l=1,p=0}^{B,I-1} \cup \{*\}$ denotes the symbol at frame t, which can be either a pair (l,p) of NV l and pitch p or a blank label $*$. $\mathcal{M}(\bar{\mathbf{Y}})$ is a reduction operator that annexes repeated symbols and removes all blank symbols from $\bar{\mathbf{Y}}$. For example, if

$$\bar{\mathbf{Y}} = ((2,61),(2,61),*,*,(2,66),*,*,(4,64),*,*,*,*,*)\,, \tag{15}$$

then

$$\mathcal{M}(\bar{\mathbf{Y}}) = ((2,61),(2,66),(4,64))\,. \tag{16}$$

A DNN is trained such that the posterior probability $p(\mathbf{Y}|\mathbf{X})$ of the ground-truth sequence $\mathbf{Y}$ is maximized. It is computed by accumulating the posterior probabilities $p(\bar{\mathbf{Y}}|\mathbf{X})$ of all $\bar{\mathbf{Y}}$ that can be reduced to $\mathbf{Y}$ as follows:

$$p(\mathbf{Y}|\mathbf{X}) = \sum_{\bar{\mathbf{Y}} \in \mathcal{M}^{-1}(\mathbf{Y})} p(\bar{\mathbf{Y}}|\mathbf{X}), \tag{17}$$

where $\mathcal{M}^{-1}(\mathbf{Y})$ represents the set of $\bar{\mathbf{Y}}$ that can be reduced to $\mathbf{Y}$. Assuming the conditional independence over frames, we have

$$p(\bar{\mathbf{Y}}|\mathbf{X}) = \prod_{t=1}^{T} p(\bar{y}_t|\mathbf{X}), \tag{18}$$

where $p(\bar{y}_t|\mathbf{X})$ is the local posterior probability given by

$$p(\bar{y}_t|\mathbf{X}) = \begin{cases} \phi_t^*, & \bar{y}_t = *, \\ (1-\phi_t^*)\phi_{tl}^L \phi_{tp}^P, & \bar{y}_t = (l,p). \end{cases} \tag{19}$$

Here ϕ_{tl}^L is the posterior probability of NV l, ϕ_{tp}^P is that of pitch p, and ϕ_t^* is the probability of $\bar{y}_t = *$ at frame t. As shown in the left example of Figure 2, the DNN is used for jointly estimating $\boldsymbol{\phi}^L = \left(\phi_{tl}^L\right)_{t=1,l=1}^{T,B}$, $\boldsymbol{\phi}^P = \left(\phi_{tp}^P\right)_{t=1,p=0}^{T,I-1}$, and $\boldsymbol{\phi}^* = (\phi_t^*)_{t=1}^T$ from $\mathbf{X}$ as follows:

$$(\boldsymbol{\phi}^L, \boldsymbol{\phi}^P, \boldsymbol{\phi}^*) = \mathrm{DNN}(\mathbf{X}). \tag{20}$$

4.2.2 MP-Based Representation

The CTC-based training using the acoustic features $\mathbf{X}$ and the ground-truth MP-based representation $\mathbf{Y} = (\mathbf{B}, \mathbf{P})$ can be performed in the same way as described in Section 4.2.1. The frame-level sequence $\bar{\mathbf{Y}} = (\bar{y}_t)_{t=1}^T$ is defined as $\bar{y}_t \in \{(b,p)\}_{b=0,p=0}^{B-1,I-1} \cup \{*\}$. The local posterior probability $p(\bar{y}_t|\mathbf{X})$ is given by

$$p(\bar{y}_t|\mathbf{X}) = \begin{cases} \phi_t^*, & \bar{y}_t = *, \\ (1-\phi_t^*)\phi_{tb}^B \phi_{tp}^P, & \bar{y}_t = (b,p). \end{cases} \tag{21}$$

As shown in the right example of Figure 2, the DNN is used for jointly estimating $\boldsymbol{\phi}^B = \left(\phi_{tb}^B\right)_{t=1,b=0}^{T,B-1}$, $\boldsymbol{\phi}^P = \left(\phi_{tp}^P\right)_{t=1,p=0}^{T,I-1}$, and $\boldsymbol{\phi}^* = (\phi_t^*)_{t=1}^T$ as follows:

$$(\boldsymbol{\phi}^B, \boldsymbol{\phi}^P, \boldsymbol{\phi}^*) = \mathrm{DNN}(\mathbf{X}). \tag{22}$$

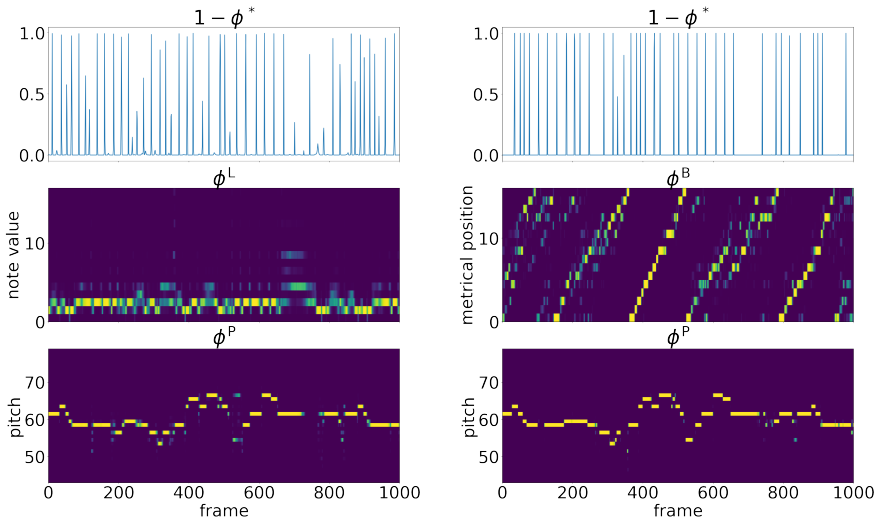

Figure 2: The frame-level posterior probabilities of symbols estimated by the DNN: The NV-based representation (left) and MP-based representation (right). For readability, the non-blank probability $1-\phi_t^*$ is shown instead of ϕ_t^*.

A notable difference between the left (NV-based) and right (MP-based) columns in Figure 2 is the middle panels where rhythmic information is estimated. In the output of the MP-based representation, period structure of metrical positions is visible and indicates local tempo values, where such structure is not visible in the output of the NV-based representation.

4.3 Network Architecture

The DNN used in this study is based on an encoder-decoder architecture (Figure 3).

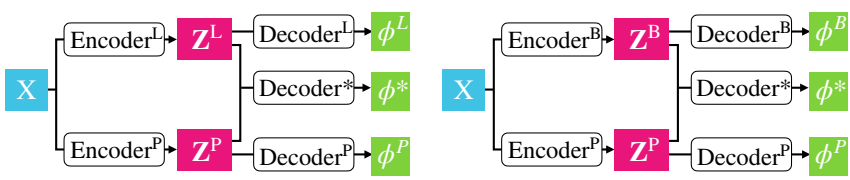

Figure 3: Network architectures for the NV-based representation (left) and the MP-based representation (right).

4.3.1 NV-Based Representation

Wide-band frequency spectral structure and long-term temporal structure are useful clues for pitch and rhythm estimation, respectively. To estimate the pitch at each frame, it would be sufficient to focus on the instantaneous features of the singing voice around the frame. To estimate the NV of a note, in contrast, it is necessary to aggregate information of periodic structure in audio features over multiple measures from the accompaniment part of music. We thus use two separate encoders for extracting latent NV features $\mathbf{Z}^L$ and latent pitch features $\mathbf{Z}^P$ as follows:

$$\mathbf{Z}^L = \mathrm{Encoder}^L(\mathbf{X}), \tag{23}$$

$$\mathbf{Z}^P = \mathrm{Encoder}^P(\mathbf{X}). \tag{24}$$

We then compute the posterior probabilities of the NVs, pitches, and blank symbol, $\boldsymbol{\phi}^L$, $\boldsymbol{\phi}^P$, and $\boldsymbol{\phi}^*$, at the frame level. As in the encoding step, $\boldsymbol{\phi}^L$ and $\boldsymbol{\phi}^P$ are estimated from $\mathbf{Z}^L$ and $\mathbf{Z}^P$, respectively. In contrast, $\boldsymbol{\phi}^*$ is estimated from both $\mathbf{Z}^L$ and $\mathbf{Z}^P$ because the blank symbol is expected to be selected at frames with small confidence about the pitch or NV. Consequently, we have

$$\boldsymbol{\phi}^L = \mathrm{Decoder}^L(\mathbf{Z}^L), \tag{25}$$

$$\boldsymbol{\phi}^P = \mathrm{Decoder}^P(\mathbf{Z}^P), \tag{26}$$

$$\boldsymbol{\phi}^* = \mathrm{Decoder}^*(\mathbf{Z}^L, \mathbf{Z}^P). \tag{27}$$

4.3.2 MP-Based Representation

For the MP-based representation, we can use the same network architecture as the one described in Section 4.3.1. The only difference is that instead of the information about NV $\mathbf{L}$, we here extract the information about the MP $\mathbf{B}$. The NV in the NV-based representation is an integer ranging from 1 to B, whereas the MP in the MP-based representation ranges from 0 to $B-1$.

4.4 HSMM Decoding

To decode a musical score from the posterior probabilities of symbols estimated by the DNN, we use an HSMM to incorporate prior knowledge about musical scores and temporal fluctuation in human performance. We formulate this model in the following.

4.4.1 Note-Level Model Formulation

As a naive decoding method, we can estimate the frame- and tatum-level sequences $\hat{\bar{\mathbf{Y}}}$ and $\hat{\mathbf{Y}}$ as follows:

$$\hat{\bar{\mathbf{Y}}} = \operatorname*{argmax}_{\bar{\mathbf{Y}}} \; p(\bar{\mathbf{Y}}|\mathbf{X}), \tag{28}$$

$$\hat{\mathbf{Y}} = \mathcal{M}(\hat{\bar{\mathbf{Y}}}), \tag{29}$$

where $p(\bar{\mathbf{Y}}|\mathbf{X})$ is calculated from the DNN output (see (18)). This method can be applied to both NV- and MP-based representations. The problem with this method is that the output sequence often violates the regularities seen in real music data, such as those in pitch and MP sequences [22] and in tempo changes [19]. This problem can be addressed by incorporating a model for musical scores and a model of temporal fluctuations in musical performance, similarly as the use of a language model in the CTC-based ASR method [8].

We propose a refined decoding method that combines a note-level musical score model based on an HMM and a frame-level tempo model based on an HSMM. Consider the MP-based note-level score representation $\mathbf{Y} = (\mathbf{B}, \mathbf{P})$ as described in Section 2.2.2. To construct the model, we represent a score with a sequence of notes $y_n = (b_n, s_n, p_n)$, where b_n is the MP, s_n is the local key, and p_n is the pitch. Here we introduced the key variables s_n corresponding to pitches p_n, which are treated as hidden variables. These variables represent the musical scale structure, which is a useful clue for determining the pitches. For simplicity, we ignore the difference between the major and minor scales, and consider that the C major key is identical to the A minor key. Thus, we have $s_n \in \{0, \ldots, 11\}$, where s_n represents the pitch class of the major-key tonic. We use a formulation similar to that in Nishikimi *et al.* [22], and the HMM-based language model can be formulated as

$$p(\mathbf{B}, \mathbf{S}, \mathbf{P}) = p(b_1)p(s_1)p(p_1|s_1) \prod_{n=2}^{N} p(b_n|b_{n-1})p(s_n|s_{n-1})p(p_n|p_{n-1}, s_n). \tag{30}$$

We formulate the probabilities related to pitches so that they are symmetric under transpositions for efficient learning of the parameters. For this purpose, we introduce the key-relative pitch class of pitch p in key s as

$$\deg(s, p) = \begin{cases} (p - s) \bmod 12, & p < 128, \\ 12, & p = 128. \end{cases} \tag{31}$$

Then, the initial probabilities are parametrized as

$$p(b_1 = b) = \pi^B_b, \tag{32}$$

$$p(s_1 = s) = \pi^S_s, \tag{33}$$

$$p(p_1 = p|s_1 = s) \propto \pi^P_{\deg(s,p)}, \tag{34}$$

and the transition probabilities as

$$p(b_n = b'|b_{n-1} = b) = \xi^B_{bb'}, \tag{35}$$

$$p(s_n = s'|s_{n-1} = s) = \xi^S_{(s-s') \bmod 12}, \tag{36}$$

$$p(p_n = p'|p_{n-1} = p, s_n = s) \propto \xi^P_{\deg(s,p)\deg(s,p')}. \tag{37}$$

The parameters $\boldsymbol{\pi}^B, \boldsymbol{\pi}^S, \boldsymbol{\pi}^P, \boldsymbol{\xi}^B, \boldsymbol{\xi}^S, \boldsymbol{\xi}^P$ can be learned from the ground-truth score data of the training data. Specifically, we extract the pitches, onset score times, and key signatures from the score data and compute the local key of each note using a standard mapping from a key signature to the key tonic. Using the method of maximum-likelihood estimation, we can then learn those parameters by computing the relative frequencies of the corresponding musical events in the data. For example, π^B_b is the relative frequency of the MP b of the first note of a song. These parameters can then be used in the frame-level HSMM decoder explained in Section 4.4.2.

4.4.2 Frame-Level Model Formulation

To estimate the onset score times of musical notes from frame-level DNN outputs, it is necessary to incorporate in the decoding process a tempo model [19], which describes the temporal fluctuation in human performance. For this purpose, we formulate a frame-level HSMM by introducing the local tempos $\mathbf{V} = (v_n)_{n=1}^N$, where $v_n \in \mathbb{R}_{>0}$ is defined as the ratio between the duration (measured in time frames) and NV of the n-th note. Following the model in Nakamura *et al.* [19], the tempo model is represented as a Markov model of local tempos. In this model, we consider that the *logarithm* of the tempo, instead of the tempo itself, follows a normal distribution to account for the covariance under a scaling transformation. Introducing the logarithmic tempos $\mathbf{U} = (u_n)_{n=1}^N = (\ln v_n)_{n=1}^N$, the tempo model is given as follows:

$$p(u_1) = \mathcal{N}\left(u_1; \mu_u^{\mathrm{ini}}, (\sigma_u^{\mathrm{ini}})^2\right), \tag{38}$$

$$p(u_n|u_{n-1}) = \mathcal{N}\left(u_n; u_{n-1}, \sigma_u^2 \Delta(b_{n-1}, b_n)^2\right). \tag{39}$$

Here, Δ is defined in (11), $\mathcal{N}(x; \mu, \sigma^2) = \exp(-(x-\mu)^2/2\sigma^2)/\sqrt{2\pi\sigma^2}$ denotes the normal distribution with mean μ and standard deviation σ, μ_u^{ini} represents the mean initial logarithmic tempo, σ_u^{ini} the standard deviation of the initial logarithmic tempos, and σ_u the standard deviation of logarithmic tempo changes in a tatum unit. The standard deviation in the right-hand side (RHS) of (39) is multiplied by the NV $\Delta(b_{n-1}, b_n)$ to account for the larger accumulation of tempo changes during a longer note [17]. With the tempo variable, the expectation value of the duration d_n of note n is given by $\Delta(b_n, b_{n+1})v_n = \Delta(b_n, b_{n+1})\exp(u_n)$. The effect of the timing fluctuation

due to motor noise and the quantization errors in signal processing can be represented by a normal distribution

$$p(d_n|b_n, b_{n+1}, u_n) \propto \mathcal{N}\left(d_n; \Delta(b_n, b_{n+1}) \exp(u_n), \sigma_c^2/L_f^2\right), \tag{40}$$

where L_f = (hop size)/(sampling rate) is the length of a frame measured in seconds, and σ_c represents the standard deviation for the timing fluctuation measured in seconds. Note that each d_n takes an integer value and the RHS of (40) should be properly normalized. The last frame t_n of note n can be calculated as

$$t_n = \begin{cases} 0, & n = 0, \\ \sum_{n'=1}^{n} d_{n'}, & n = 1, \ldots, N. \end{cases} \tag{41}$$

We then define the frame-level variables, the MP $\bar{\mathbf{B}} = \left(\bar{b}_t\right)_{t=0}^{T}$, key $\bar{\mathbf{S}} = (\bar{s}_t)_{t=1}^{T}$, pitch $\bar{\mathbf{P}} = (\bar{p}_t)_{t=1}^{T}$, logarithmic tempo $\bar{\mathbf{U}} = (\bar{u}_t)_{t=0}^{T}$, and counter $\bar{\mathbf{C}} = (\bar{c}_t)_{t=1}^{T}$ as follows: if $t_{n-1} < t \leq t_n$, then

$$\bar{b}_t = b_{n+1}, \tag{42}$$
$$\bar{s}_t = s_n, \tag{43}$$
$$\bar{p}_t = p_n, \tag{44}$$
$$\bar{u}_t = u_{n+1}, \tag{45}$$
$$\bar{c}_t = t_n - t. \tag{46}$$

The counter variable $\bar{c}_t$ represents the remaining number of frames within the current note. Note that in (42) we assign the MP of the *next* note because it is needed to calculate the NV of the current note at its onset frame. Similarly in (45), we assign the local tempo of the next note so that the probabilistic model for the counter variable can be represented by a Markov model (see (59)). Accordingly, to represent the onset MP and tempo of the first note, we introduce $\bar{b}_0 = b_1$ and $\bar{u}_0 = u_1$. Table 1 shows an example of these variables.

Consider the frame-level input and output features $\mathbf{X}$ and $\bar{\mathbf{Y}}$ described in Section 4.2.2. We formulate the generative process of $\bar{\mathbf{B}}, \bar{\mathbf{S}}, \bar{\mathbf{P}}, \bar{\mathbf{U}}, \bar{\mathbf{C}}, \bar{\mathbf{Y}}$, and $\mathbf{X}$ as

$$p(\mathbf{X}, \bar{\mathbf{B}}, \bar{\mathbf{S}}, \bar{\mathbf{P}}, \bar{\mathbf{U}}, \bar{\mathbf{C}}, \bar{\mathbf{Y}}) = p(\mathbf{X}|\bar{\mathbf{Y}}) p(\bar{\mathbf{B}}, \bar{\mathbf{S}}, \bar{\mathbf{P}}, \bar{\mathbf{U}}, \bar{\mathbf{C}}, \bar{\mathbf{Y}}). \tag{47}$$

Here, $p(\bar{\mathbf{B}}, \bar{\mathbf{S}}, \bar{\mathbf{P}}, \bar{\mathbf{U}}, \bar{\mathbf{C}}, \bar{\mathbf{Y}})$ represents the language model for the frame-level variables, which is given as

Table 1: An example of the frame-level variables for the MP-based representation. Instead of $\bar{u}_t$, we show the values of tempos $\bar{v}_t = \exp(\bar{u}_t)$. The bold fonts indicate that $\bar{y}_t$ is determined by $\bar{b}_{t-1}$, $\bar{p}_t$, and $\bar{c}_{t-1}$.

t	0	1	2	3	4	5	6	7	8	9	10	11	...
$\bar{b}_t$	**0**	2	2	2	2	2	2	**2**	3	3	3	**3**	...
$\bar{s}_t$		1	1	1	1	1	1	1	1	1	1	1	...
$\bar{p}_t$		**62**	62	62	62	62	62	62	**59**	59	59	59	...
$\exp(\bar{u}_t)$	3.5	4	4	4	4	4	4	4	4.1	4.1	4.1	4.1	...
$\bar{c}_t$	**0**	6	5	4	3	2	1	**0**	3	2	1	**0**	...
$\bar{y}_t$		**(0, 62)**	*	*	*	*	*	*	**(2, 59)**	*	*	*	...

$$
\begin{aligned}
p(\bar{\mathbf{B}}, \bar{\mathbf{S}}, \bar{\mathbf{P}}, \bar{\mathbf{U}}, \bar{\mathbf{C}}, \bar{\mathbf{Y}}) = {} & p(\bar{b}_0, \bar{b}_1) p(\bar{s}_1) p(\bar{p}_1 | \bar{s}_1) p(\bar{u}_0, \bar{u}_1 | \bar{b}_0, \bar{b}_1) p(\bar{c}_0 | \bar{b}_0, \bar{b}_1, \bar{u}_0) \\
& \cdot \prod_{t=2}^{T} p(\bar{b}_t | \bar{b}_{t-1}, \bar{c}_{t-1}) p(\bar{s}_t | \bar{s}_{t-1}, \bar{c}_{t-1}) \\
& \cdot \prod_{t=2}^{T} p(\bar{p}_t | \bar{p}_{t-1}, \bar{s}_t, \bar{c}_{t-1}) p(\bar{u}_t | \bar{b}_{t-1}, \bar{b}_t, \bar{u}_{t-1}, \bar{c}_{t-1}) \\
& \cdot \prod_{t=1}^{T} p(\bar{c}_t | \bar{b}_{t-1}, \bar{b}_t, \bar{u}_{t-1}, \bar{c}_{t-1}) p(\bar{y}_t | \bar{b}_{t-1}, \bar{p}_t, \bar{c}_{t-1}). \qquad (48)
\end{aligned}
$$

The initial and transition probabilities of the MPs are

$$
p(\bar{b}_0, \bar{b}_1) = p(\bar{b}_0) p(\bar{b}_1 | \bar{b}_0) = \pi^B_{\bar{b}_0} \xi^B_{\bar{b}_0 \bar{b}_1}, \qquad (49)
$$

$$
p(\bar{b}_t | \bar{b}_{t-1}, \bar{c}_{t-1}) = \begin{cases} \delta(\bar{b}_{t-1}, \bar{b}_t), & \bar{c}_{t-1} > 0, \\ \xi^B_{\bar{b}_{t-1} \bar{b}_t}, & \bar{c}_{t-1} = 0, \end{cases} \qquad (50)
$$

where δ stands for the Kronecker delta:

$$
\delta(x, y) = \begin{cases} 1, & x = y, \\ 0, & x \neq y. \end{cases} \qquad (51)
$$

The initial and transition probabilities of the keys are

$$
p(\bar{s}_1) = \pi^S_{\bar{s}_1}, \qquad (52)
$$

$$
p(\bar{s}_t | \bar{s}_{t-1}, \bar{c}_{t-1}) = \begin{cases} \delta(\bar{s}_{t-1}, \bar{s}_t), & \bar{c}_{t-1} > 0, \\ \xi^S_{(\bar{s}_{t-1} - \bar{s}_t) \bmod 12}, & \bar{c}_{t-1} = 0. \end{cases} \qquad (53)
$$

The initial and transition probabilities of the pitches given the current keys are

$$
p(\bar{p}_1 | \bar{s}_1) \propto \pi^P_{\deg(\bar{s}_1, \bar{p}_1)}, \qquad (54)
$$

$$
p(\bar{p}_t | \bar{p}_{t-1}, \bar{s}_t, \bar{c}_{t-1}) \propto \begin{cases} \delta(\bar{p}_{t-1}, \bar{p}_t), & \bar{c}_{t-1} > 0, \\ \xi^P_{\deg(\bar{s}_t, \bar{p}_{t-1}) \deg(\bar{s}_t, \bar{p}_t)}, & \bar{c}_{t-1} = 0. \end{cases} \qquad (55)
$$

The initial and transition probabilities of the logarithmic tempos are

$$\begin{aligned} p(\bar{u}_0, \bar{u}_1|\bar{b}_0, \bar{b}_1) &= p(\bar{u}_0)p(\bar{u}_1|\bar{u}_0, \bar{b}_0, \bar{b}_1) \\ &= \mathcal{N}\left(\bar{u}_0; \mu_u^{\text{ini}}, (\sigma_u^{\text{ini}})^2\right)\mathcal{N}\left(\bar{u}_1; \bar{u}_0, \sigma_u^2\Delta(\bar{b}_0, \bar{b}_1)^2\right), \end{aligned} \tag{56}$$

$$p(\bar{u}_t|\bar{b}_{t-1}, \bar{b}_t, \bar{u}_{t-1}, \bar{c}_{t-1}) = \begin{cases} \delta(\bar{u}_t - \bar{u}_{t-1}), & \bar{c}_{t-1} > 0, \\ \mathcal{N}\left(\bar{u}_t; \bar{u}_{t-1}, \sigma_u^2\Delta(\bar{b}_{t-1}, \bar{b}_t)^2\right), & \bar{c}_{t-1} = 0, \end{cases} \tag{57}$$

where $\delta(x)$ denotes the Dirac delta function for a continuous variable. The initial and transition probabilities of the counters are

$$p(\bar{c}_0|\bar{b}_0, \bar{b}_1, \bar{u}_0) = \text{Uniform}[0, \Delta(\bar{b}_0, \bar{b}_1)\exp(\bar{u}_0) - 1 + \sigma_c^{\text{ini}}/L_f], \tag{58}$$

$$p(\bar{c}_t|\bar{b}_{t-1}, \bar{b}_t, \bar{u}_{t-1}, \bar{c}_{t-1}) \propto \begin{cases} \delta(\bar{c}_{t-1} - 1, \bar{c}_t), & \bar{c}_{t-1} > 0, \\ \mathcal{N}\left(\bar{c}_t; \Delta(\bar{b}_{t-1}, \bar{b}_t)\exp(\bar{u}_{t-1}) - 1, \sigma_c^2/L_f^2\right), & \bar{c}_{t-1} = 0. \end{cases} \tag{59}$$

In the initial probability, we allow that the audio signal can start in the middle of the first note and that the note's maximum duration is given as its expectation value added by some amount of deviation represented by $\sigma_c^{\text{ini}}/L_f$. Last, the emission probabilities of framewise output labels given the hidden variables are

$$p(\bar{y}_t|\bar{b}_{t-1}, \bar{p}_t, \bar{c}_{t-1}) = \begin{cases} \delta(\bar{y}_t, *), & \bar{c}_{t-1} > 0, \\ \delta(\bar{y}_t, (\bar{b}_{t-1}, \bar{p}_t)), & \bar{c}_{t-1} = 0. \end{cases} \tag{60}$$

In (47), $p(\mathbf{X}|\bar{\mathbf{Y}})$ represents the emission probabilities of the observed spectrograms given $\bar{\mathbf{Y}}$, which is given as

$$p(\mathbf{X}|\bar{\mathbf{Y}}) = \prod_{t=1}^{T} p(\mathbf{x}_t|\bar{y}_t), \tag{61}$$

According to Bayes' theorem, the framewise emission probability can be written as

$$p(\mathbf{x}_t|\bar{y}_t) = \frac{p(\bar{y}_t|\mathbf{x}_t)p(\mathbf{x}_t)}{p(\bar{y}_t)} \propto \frac{p(\bar{y}_t|\mathbf{x}_t)}{p(\bar{y}_t)}. \tag{62}$$

We assume that

$$p(\bar{y}_t) = \begin{cases} \mu_*, & \bar{y}_t = *, \\ (1 - \mu_*)\pi_b^B\pi_p^P, & \bar{y}_t = (b, p), \end{cases} \tag{63}$$

where μ_* is the prior probability of the blank label. $p(\bar{y}_t|\mathbf{x}_t)$ can be estimated with the DNN described in Section 4.2.2 as

$$p(\bar{y}_t|\mathbf{x}_t) = \begin{cases} \phi_t^*, & \bar{y}_t = *, \\ (1 - \phi_t^*)\phi_{tb}^B\phi_{tp}^P, & \bar{y}_t = (b, p). \end{cases} \tag{64}$$

4.4.3 Viterbi Algorithm

With the probabilistic model described above, we can estimate the output sequence $\hat{\mathbf{Y}}$ by first estimating

$$\hat{\bar{\mathbf{B}}}, \hat{\bar{\mathbf{S}}}, \hat{\bar{\mathbf{P}}}, \hat{\bar{\mathbf{U}}}, \hat{\bar{\mathbf{C}}}, \hat{\bar{\mathbf{Y}}} = \operatorname{argmax} p(\mathbf{X}, \bar{\mathbf{B}}, \bar{\mathbf{S}}, \bar{\mathbf{P}}, \bar{\mathbf{U}}, \bar{\mathbf{C}}, \bar{\mathbf{Y}}), \tag{65}$$

which can be solved efficiently with the Viterbi algorithm.

Since the Viterbi algorithm cannot deal with continuous variables, we discretize the logarithmic tempo $\bar{u}_t$ into N_u equally spaced values in the range $[\ln v_{\min}, \ln v_{\max}]$, where $v_{\min}$ and $v_{\max}$ are the minimal and maximal tempos. The initial and transition probabilities of $\bar{u}_t$ are accordingly changed to the discrete version in the Viterbi algorithm. Additionally, we also limit the possible values of pitches to $\{p_{\min}, p_{\min}+1, \ldots, p_{\max}, 128\}$ in order to decrease the computational cost, where $p_{\min}$ and $p_{\max}$ are the minimal and maximal pitches used in vocal melodies.

Denote $q_t = (\bar{b}_t, \bar{s}_t, \bar{p}_t, \bar{u}_t, \bar{c}_t, \bar{y}_t)$. We calculate the Viterbi variables in the forward process as

$$\begin{aligned}
\omega_1(q_0, q_1) = {} & p(\bar{b}_0, \bar{b}_1)^{\gamma_b} p(\bar{s}_1)^{\gamma_p} p(\bar{p}_1|\bar{s}_1)^{\gamma_p} p(\bar{u}_0, \bar{u}_1|\bar{b}_0, \bar{b}_1) \\
& \cdot p(\bar{c}_0|\bar{b}_0, \bar{b}_1, \bar{u}_0) p(\bar{c}_1|\bar{b}_0, \bar{b}_1, \bar{u}_0, \bar{c}_0) \\
& \cdot p(\bar{y}_1|\bar{b}_0, \bar{p}_1, \bar{c}_0) p(\mathbf{x}_1|\bar{y}_1), \qquad (66) \\
\omega_t(q_t) = {} & \max_{q_{t-1}} \left[\omega_{t-1}(q_{t-1}) p(\bar{b}_t|\bar{b}_{t-1}, \bar{c}_{t-1})^{\gamma_b} \right. \\
& \cdot p(\bar{s}_t|\bar{s}_{t-1}, \bar{c}_{t-1})^{\gamma_p} p(\bar{p}_t|\bar{p}_{t-1}, \bar{s}_t, \bar{c}_{t-1})^{\gamma_p} \\
& \cdot p(\bar{u}_t|\bar{b}_{t-1}, \bar{b}_t, \bar{u}_{t-1}, \bar{c}_{t-1}) p(\bar{c}_t|\bar{b}_{t-1}, \bar{b}_t, \bar{u}_{t-1}, \bar{c}_{t-1}) \\
& \left. \cdot p(\bar{y}_t|\bar{b}_{t-1}, \bar{p}_t, \bar{c}_{t-1}) p(\mathbf{x}_t|\bar{y}_t) \right], \qquad (67) \\
\mathrm{pre}_t(q_t) = {} & \operatorname*{argmax}_{q_{t-1}} \left[\omega_{t-1}(q_{t-1}) p(\bar{b}_t|\bar{b}_{t-1}, \bar{c}_{t-1})^{\gamma_b} \right. \\
& \cdot p(\bar{s}_t|\bar{s}_{t-1}, \bar{c}_{t-1})^{\gamma_p} p(\bar{p}_t|\bar{p}_{t-1}, \bar{s}_t, \bar{c}_{t-1})^{\gamma_p} \\
& \cdot p(\bar{u}_t|\bar{b}_{t-1}, \bar{b}_t, \bar{u}_{t-1}, \bar{c}_{t-1}) p(\bar{c}_t|\bar{b}_{t-1}, \bar{b}_t, \bar{u}_{t-1}, \bar{c}_{t-1}) \\
& \left. \cdot p(\bar{y}_t|\bar{b}_{t-1}, \bar{p}_t, \bar{c}_{t-1}) p(\mathbf{x}_t|\bar{y}_t) \right], \qquad (68)
\end{aligned}$$

where weights γ_b and γ_p have been introduced to balance the influence of the language model. These parameters can in principle be optimized using validation data and in this study, due to the lack of large data, we conduct a rough grid search on the test data and examine how the transcription performance changes by changing their values (see Section 5.2.1). The optimal path can then be obtained by the backward process

$$\hat{q}_T = \operatorname*{argmax}_{q_T} \omega_T(q_T), \tag{69}$$

$$\hat{q}_t = \mathrm{pre}_{t+1}(\hat{q}_{t+1}). \tag{70}$$

We also apply a postprocessing method to suppress unexpected extra notes in the output. In a song, there are parts where no singing voice exists, such as the intro, interlude, and outro. These parts are usually represented as long rests on a musical score, but it is difficult for the present decoder to output such long rests. We assume that high probabilities of the blank symbol indicate these rests, and substitute the rest for the original decoded pitch if a frame has a high blank probability, that is,

$$\bar{p}'_t = \begin{cases} \bar{p}_t, & \phi^*_t \leq \theta_*, \\ 128, & \phi^*_t > \theta_*, \end{cases} \tag{71}$$

$$\bar{y}'_t = \begin{cases} *, & \bar{c}_{t-1} > 0, \\ (\bar{b}_{t-1}, \bar{p}'_t), & \bar{c}_{t-1} = 0, \end{cases} \tag{72}$$

where the threshold θ_* is a hyperparameter. We obtain the final decoding result $\hat{\mathbf{Y}}$ by annexing the repeated symbols and deleting the blank labels from the sequence $(\bar{y}'_t)$.

5 Evaluation

This section reports comparative experiments using real popular music songs.

5.1 Experimental Conditions

We explain the training and test data, network configuration, compared methods, and evaluation measures.

5.1.1 Dataset

To evaluate the methods in a practical setup, we made an in-house dataset consisting of 343 Japanese popular (JPOP) songs, which was randomly split into a training set of 308 songs and a test set of 35 songs. The musical scores of those songs were transcribed in the MusicXML format by experts and the audio recordings were obtained from the original CDs. We also used in the test phase a publicly available dataset named "RWC Music Database: Popular Music" [6]. Following a previous study [22], we used 12 songs (Nos. 7, 8, 13, 18, 20, 47, 63, 79, 80, 84, 90, and 100) with the time signature of 4/4. The musical scores of these songs were also transcribed by experts. Note that datasets (e.g., MIR-ST500 [29]) commonly used for audio-to-MIDI AST cannot be used in this study due to the lack of ground-truth musical scores.

To reduce the computational cost, the music signal of each song was resampled at 22050 Hz and split into segments of 8 seconds. The corresponding

ground-truth score was also split accordingly by performing audio-to-MIDI alignment based on dynamic time warping. Note that this procedure was just for preparing the training data used for end-to-end learning, where the audio signal (input) and score (output) of each segment were given in a non-aligned manner in the training phase. The magnitude spectrogram of each song was obtained with short-time Fourier transform (STFT) with a window of 2048 samples and a hop length of 256 samples ($L_f = 0.01161$ s). The mel spectrogram was then obtained with 128 mel filter banks.

5.1.2 Network Configuration

Our method is based on a CRNN, a combination of a CNN-based encoder and an RNN-based decoder, with the NV- or MP-based representation (Figure 3). Encoder^L, Encoder^B, and Encoder^P were implemented with a CNN consisting of five layers with 64, 32, 32, 32, and 32 channels and a kernel sizes of 5, 5, 3, 3, and 3, followed by a fully-connected layer that outputs a 256-dimensional latent vector at each frame. Decoder^L, Decoder^B, Decoder^P, and Decoder* were implemented with an RNN consisting of two bidirectional long short-term memory (LSTM) layers that had 256 channels in the hidden spaces. The CRNN was trained by an Adam optimizer with a learning rate of 1×10^{-4}.

For the HSMM used for decoding, the initial and transition probabilities $\boldsymbol{\pi}^B, \boldsymbol{\pi}^S, \boldsymbol{\pi}^P, \boldsymbol{\xi}^B, \boldsymbol{\xi}^S$, and $\boldsymbol{\xi}^P$ were learned from the ground-truth scores in the training data. The note pitches were limited between $p_{\min} = 43$ (G2) and $p_{\max} = 79$ (G5), the tempos were between $v_{\min} = 5.38$ (240 bpm) and $v_{\max} = 25.84$ (50 bpm), and the number of discretized logarithmic tempos was $N_u = 35$. The mean value of the initial logarithmic tempo was $\mu_u^{\text{ini}} = \ln[(v_{\min} + v_{\max})/2]$, the standard deviations of the tempo transition model were $\sigma_u = 0.0033$ and $\sigma_c = 0.03$, the initial standard deviations were $\sigma_u^{\text{ini}} = 3\sigma_v$, $\sigma_c^{\text{ini}} = 3\sigma_c$, the prior probability of the blank label was $\mu_* = 0.9$, the balancing weights were $\gamma_b = 0.6$ and $\gamma_p = 0.2$, and the threshold of the postprocessing method (introduced in (71)) was $\theta_* = 0.5$.

5.1.3 Compared Methods

We tested two variants of the proposed method trained on the JPOP dataset; one with the NV-based representation and the other with the MP-based representation. For the NV-based representation, we used the naive decoding. For the MP-based representation, we tested both the naive decoding and the HSMM decoding (default). As a baseline, we tested a cascading method that sequentially uses an audio-to-MIDI singing transcription method [11] and a rhythm transcription method [27]. We also tested the state-of-the-art audio-to-score AST method based on a CRNN-HSMM hybrid model trained

on the RWC dataset [22]. Additionally, we compared the CTC-based AST method with the tatum-based representation proposed in our previous work [3].

5.1.4 Evaluation Measures

We evaluated the estimated scores with the edit-distance-based metrics called MUSTER [9, 16], which is similar to the word error rate (WER) used for ASR. The metrics count five exclusive types of errors: pitch error rate E_{p}, missing note rate E_{m}, extra note rate E_{e}, onset time error rate E_{on}, and offset time error rate E_{off}. The E_{p}, E_{m}, and E_{e} are similar to the substitution, deletion, and insertion error rates in the WER metric.

5.2 Experimental Results

We report the performances of the compared methods and discuss the efficacy of the refined representation.

5.2.1 Performance Comparison

Table 2 shows that the proposed MP-based model had a lower mean error rate than the NV-based model and the tatum-level model [3] for both datasets. The MP-based method outperformed the CRNN-HSMM hybrid model [22] on the JPOP dataset, but had a higher mean error rate on the RWC test. This may have been caused by the difference of the training data. The training data of the CRNN-HSMM hybrid model was part of the RWC dataset, while the MP-based model used part of the JPOP dataset for training. The models may have adapted to their own training data and performed better on the test data with similar characteristics to the training data. These results show that the proposed model was no worse than the CRNN-HSMM hybrid model. Compared to the CRNN-HSMM hybrid model, which requires the time-aligned beat times as the ground-truth data for training, our proposed method can be trained in an end-to-end manner and potentially utilize large non-aligned data. The cascading method had the highest mean error rate on both test datasets.

5.2.2 Efficacy of the HSMM Decoder

To evaluate the effectiveness of the HSMM decoder, we also tested a method using the naive decoder in (28) and (29). The result in Table 3 shows that without the HSMM decoder, the mean error rate was higher on both datasets. We found that the accuracy of recognizing the onset times was significantly improved by the HSMM. This clearly shows the efficacy of the decoder.

Table 2: Comparison of edit-distance-based error rates (%) of different methods.

Test data	Method	E_{p}	E_{m}	E_{e}	E_{on}	E_{off}	Mean
JPOP	Cascade ([11]+[27])	9.0	47.8	14.2	47.4	31.2	29.9
	CRNN-HSMM [22]	9.2	21.0	11.5	31.2	24.5	19.5
	Tatum-level [3]	8.8	23.4	14.1	35.1	26.4	21.6
	NV-based	**6.9**	24.6	**7.5**	37.3	27.3	20.7
	MP-based	10.3	**14.1**	11.0	**24.7**	**23.7**	**16.8**
RWC	Cascade ([11]+[27])	12.7	16.5	15.6	40.6	29.5	23.0
	CRNN-HSMM [22]	**7.9**	9.6	14.5	**22.5**	**21.6**	**15.2**
	Tatum-level [3]	9.4	13.3	18.9	32.3	26.9	20.2
	NV-based	9.1	12.1	**11.7**	35.5	24.0	18.5
	MP-based	10.6	**8.8**	15.9	24.5	27.1	17.4

Table 3: Comparison of edit-distance-based error rates (%) of the proposed method (MP-based representation) with and without the HSMM decoder.

Test data	Method	E_{p}	E_{m}	E_{e}	E_{on}	E_{off}	Mean
JPOP	Naive decoder	**9.9**	16.7	13.6	37.5	24.2	20.4
	HSMM decoder	10.2	**14.2**	**11.0**	**24.6**	**23.5**	**16.7**
RWC	Naive decoder	**10.6**	9.2	**15.6**	36.2	**26.5**	19.6
	HSMM decoder	**10.6**	**8.5**	**15.6**	**24.8**	27.1	**17.3**

We also explored the influence of the note-level language model by changing the values of weights γ_b and γ_p. Figure 4 shows that even though a slight decrease of error rate was found when the weights were not too high, the difference in the error rate was relatively small. A possible reason for this result is that the treatment of rest notes included in the pitch language model was inappropriate, suggesting a refinement of the musical language model can further improve the result.

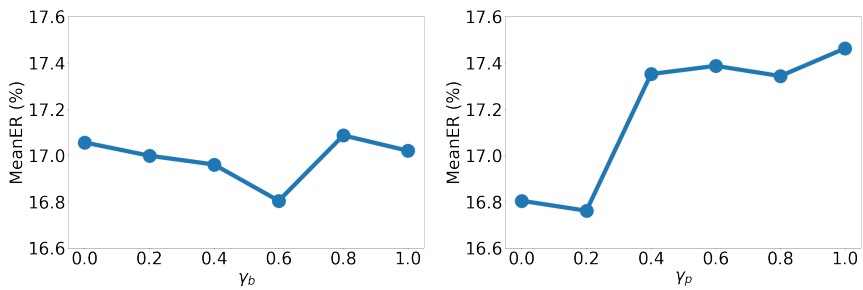

Figure 4: Dependence of the mean error rate on γ_b (left) and γ_p (right). We fix that $\gamma_p = 0$ in the left figure and $\gamma_b = 0.6$ in the right figure.

5.2.3 *Influence of Tatum Unit*

To examine the potential of the MP-based model to adapt to finer-grained tatum units, we also trained and tested the models with the 48th note tatum. By choosing this tatum unit, we expect that the model can recognize triplets, which is impossible to represent in the 16th note tatum. In a preliminary experiment, we observed that the CTC-based model with the tatum-level representation [3] failed to learn with the 48th note tatum. The reason is that the finer-grained tatum unit increased the number of symbols, and the temporal segment for each symbol became too short for the model to recognize. Therefore, it is important to see whether the proposed model still works with the finer-grained tatum unit.

Table 4 shows the results. We found that the model with the 48th note tatum had similar performance to the model with 16th note tatum in terms of MUSTER metrics. In an example shown in Figure 5, the former model indeed recognized triplet notes. However, the method incorrectly recognized triplet notes for a tied dotted rhythm in the first measure. To correctly discriminate these rhythms is a challenging problem in principle, particularly because of the significant temporal fluctuation of singing voice. These results indicated the potential of the MP-based representation for dealing with fine-grained rhythms and call for a more elaborated method for inferring rhythms, for example by utilizing the repetitive structure of music [18].

Table 4: Comparison of edit-distance-based error rates (%) of the proposed method with different tatum units. We only show the result on the JPOP test set because the RWC dataset contained no triplet notes.

Tatum unit	E_{p}	E_{m}	E_{e}	E_{on}	E_{off}	Mean
16th note	10.3	14.1	**11.0**	**24.7**	**23.7**	**16.8**
48th note	**9.6**	**14.0**	12.8	25.1	23.7	17.1

6 Conclusion and Discussion

We have proposed an end-to-end singing transcription method based on CTC. We developed the MP-based score representation, which is suited for representing the metrical structure of musical scores, and the HSMM decoder to improve the result of inference. The proposed method only requires non-aligned pairs of audio signals and musical scores for training, making it possible to potentially utilize larger datasets. The experimental results showed that the proposed method can achieve relatively high accuracy on real-world data.

Several issues are left for future work. First, we treated the rest note equivalently to pitched notes, but the result suggested that this is inappropriate.

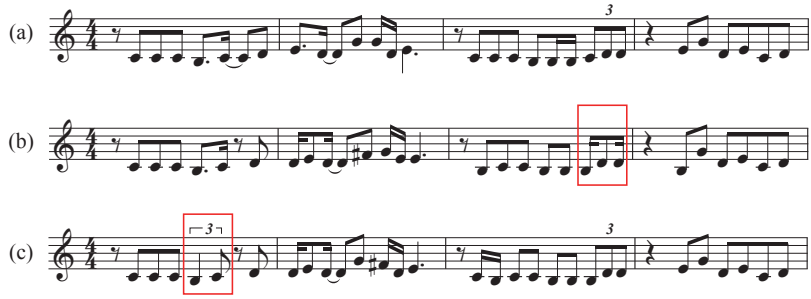

Figure 5: Estimated results with different tatum units. (a) Ground-truth score. (b) Estimated result with the 16th note tatum model. (c) Estimated result with the 48th note tatum model. Notable rhythmic errors are highlighted with boxes.

In fact, rest notes are used as special symbols to represent the parts without singing voice and they cannot be easily discriminated from the blank symbol. Thus, a proper way of dealing with rest notes should be carefully developed in the proposed framework. Second, as explained in Section 2, the proposed score representation cannot deal with notes longer than a measure. Although this problem can partly be solved by considering a wider range of note values or a longer period of metrical positions, a larger vocabulary may significantly increase the computational cost. A refined method for tying multiple notes in the postprocessing step may be necessary to deal with very long notes and rests. Finally, we found that our method often fails to detect repeated notes with the same pitch and can incorrectly split a single note into multiple notes. In fact, the difference between these two cases is ambiguous because the onsets of notes in singing voices are not always clear, especially when the same pitch is repeated.

Acknowledgment

We thank Daichi Kamakura for useful discussions. This study was partially supported by JSPS KAKENHI Nos. 21K12187, 21K02846, 22H03661, 20H00602, and 21H03572, JST PRESTO No. JPMJPR20CB, and JST FOREST No. JPMJPR226X.

References

[1] V. Arroyo, J. J. Valero-Mas, J. Calvo-Zaragoza, and A. Pertusa, "Neural Audio-to-Score Music Transcription for Unconstrained Polyphony Using Compact Output Representations", in *ICASSP*, 2022, 4603–7.

[2] S. Chowdhury, A. V. Portabella, V. Haunschmid, and G. Widmer, "Towards Explainable Music Emotion Recognition: The Route via Mid-Level Features", in *ISMIR*, 2019, 237–43.

[3] T. Deng, E. Nakamura, and K. Yoshii, "Audio-to-Score Singing Transcription Based on Joint Estimation of Pitches, Onsets, and Metrical Positions with Tatum-Level CTC Loss", in *APSIPA ASC*, 2023, 583–90.

[4] Z. Duan and B. Pardo, "Soundprism: An Online System for Score-Informed Source Separation of Music Audio", *IEEE Journal of Selected Topics in Signal Processing*, 5(6), 2011, 1205–15.

[5] S. Ewert, B. Pardo, M. Muller, and M. D. Plumbley, "Score-Informed Source Separation for Musical Audio Recordings: An Overview", *IEEE Signal Processing Magazine*, 31(3), 2014, 116–24.

[6] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, "RWC Music Database: Popular, Classical and Jazz Music Databases", in *ISMIR*, 2002, 287–8.

[7] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks", in *ICML*, 2006, 369–76.

[8] A. Graves and N. Jaitly, "Towards End-to-End Speech Recognition with Recurrent Neural Networks", in *ICML*, 2014, 1764–72.

[9] Y. Hiramatsu, E. Nakamura, and K. Yoshii, "Joint Estimation of Note Values and Voices for Audio-to-Score Piano Transcription", in *ISMIR*, 2021, 278–84.

[10] T.-H. Hsieh, L. Su, and Y.-H. Yang, "A Streamlined Encoder/Decoder Architecture for Melody Extraction", in *ICASSP*, 2019, 156–60.

[11] J.-Y. Hsu and L. Su, "VOCANO: A Note Transcription Framework for Singing Voice in Polyphonic Music", in *ISMIR*, 2021, 293–300.

[12] A. Ito, Y. Kosugi, S. Makino, and M. Ito, "A Query-by-Humming Music Information Retrieval from Audio Signals Based on Multiple F0 Candidates", in *ICALIP*, 2010, 1–5.

[13] J. W. Kim, J. Salamon, P. Li, and J. P. Bello, "CREPE: A Convolutional Representation for Pitch Estimation", in *ICASSP*, 2018, 161–5.

[14] M. Mauch, C. Cannam, R. Bittner, G. Fazekas, J. Salamon, J. Dai, J. Bello, and S. Dixon, "Computer-Aided Melody Note Transcription Using the Tony Software: Accuracy and Efficiency", in *TENOR*, 2015, 23–31.

[15] M. Mauch and S. Dixon, "PYIN: A Fundamental Frequency Estimator Using Probabilistic Threshold Distributions", in *ICASSP*, 2014, 659–63.

[16] E. Nakamura, E. Benetos, K. Yoshii, and S. Dixon, "Towards Complete Polyphonic Music Transcription: Integrating Multi-Pitch Detection and Rhythm Quantization", in *ICASSP*, 2018, 101–5.

[17] E. Nakamura, N. Ono, S. Sagayama, and K. Watanabe, "A Stochastic Temporal Model of Polyphonic MIDI Performance with Ornaments", *Journal of New Music Research*, 44(4), 2015, 287–304.

[18] E. Nakamura and K. Yoshii, "Musical Rhythm Transcription Based on Bayesian Piece-Specific Score Models Capturing Repetitions", *Information Sciences*, 572, 2021, 482–500.

[19] E. Nakamura, K. Yoshii, and S. Sagayama, "Rhythm Transcription of Polyphonic Piano Music Based on Merged-Output HMM for Multiple Voices", *TASLP*, 25(4), 2017, 794–806.

[20] R. Nishikimi, E. Nakamura, S. Fukayama, M. Goto, and K. Yoshii, "Automatic Singing Transcription Based on Encoder-Decoder Recurrent Neural Networks with a Weakly-Supervised Attention Mechanism", in *ICASSP*, 2019, 161–5.

[21] R. Nishikimi, E. Nakamura, M. Goto, K. Itoyama, and K. Yoshii, "Scale- and Rhythm-Aware Musical Note Estimation for Vocal F0 Trajectories Based on a Semi-Tatum-Synchronous Hierarchical Hidden Semi-Markov Model", in *ISMIR*, 2017, 376–82.

[22] R. Nishikimi, E. Nakamura, M. Goto, and K. Yoshii, "Audio-to-Score Singing Transcription Based on a CRNN-HSMM Hybrid Model", *APSIPA*, 10(e7), 2021, 1–13.

[23] R. Nishikimi, E. Nakamura, M. Goto, and K. Yoshii, "End-to-End Melody Note Transcription Based on a Beat-Synchronous Attention Mechanism", in *WASPAA*, 2019, 26–30.

[24] M. Rocamora, P. Cancela, and A. Pardo, "Query by Humming: Automatically Building the Database from Music Recordings", *Pattern Recognition Letters*, 36, 2014, 272–80.

[25] M. A. Román, A. Pertusa, and J. Calvo-Zaragoza, "A Holistic Approach to Polyphonic Music Transcription with Neural Networks", in *ISMIR*, 2019, 731–7.

[26] M. A. Román, A. Pertusa, and J. Calvo-Zaragoza, "An End-to-End Framework for Audio-to-Score Music Transcription on Monophonic Excerpts", in *ISMIR*, 2018, 34–41.

[27] K. Shibata, E. Nakamura, and K. Yoshii, "Non-Local Musical Statistics as Guides for Audio-to-Score Piano Transcription", *Information Sciences*, 566, 2021, 262–80.

[28] F.-R. Stöter, S. Uhlich, A. Liutkus, and Y. Mitsufuji, "Open-Unmix — A Reference Implementation for Music Source Separation", *Journal of Open Source Software*, 4(41), 2019, 1667–72.

[29] J.-Y. Wang and J.-S. R. Jang, "On the Preparation and Validation of a Large-Scale Dataset of Singing Transcription", in *ICASSP*, 2021, 276–80.

[30] J.-Y. Wang and J.-S. R. Jang, "Training a Singing Transcription Model Using Connectionist Temporal Classification Loss and Cross-Entropy Loss", *IEEE/ACM TASLP*, 31, 2023, 383–96.
[31] C. Wigington, B. Price, and S. Cohen, "Multi-Label Connectionist Temporal Classification", in *ICDAR*, 2019, 979–86.
[32] S. Yu, X. Sun, Y. Yu, and W. Li, "Frequency-Temporal Attention Network for Singing Melody Extraction", in *ICASSP*, 2021, 251–5.